PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-005725
(43)Date of publication of application : 12.01.2001

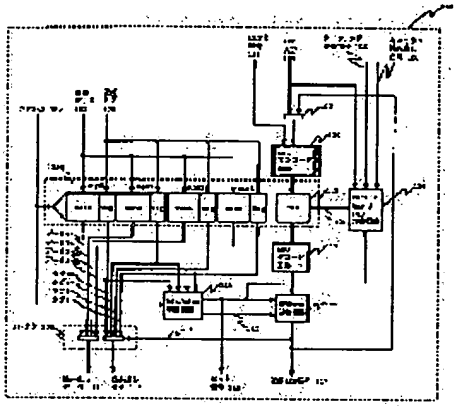
(51)Int.Cl. G06F 12/12

(21)Application number : 11-173656
(22)Date of filing : 21.06.1999

(71)Applicant : HITACHI LTD
(72)Inventor : NISHIMOTO SATORU
IIZUKA TAKAYOSHI
KAMATA EIKI

(54) CACHE STORAGE DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a cache storage device which decreases a frequency of miscaching due to a block conflict as a set associative type cache storage device.
SOLUTION: The set associative type cache memory device equipped with a memory cell part which holds data and tags, an LRU memory part representing reference history information on cache blocks, and a circuit which makes a hit/miss decision has an instruction for changing the LRU of a corresponding cache block to the block which was referred to in the remote past. When the instruction is executed, an LRU encoding circuit 110 changes the LRU of the corresponding block to the block which was referred to in the remote past.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Japanese Laid-Open Patent Application No. 2001-5725 Publication
(Lines 31 to 47)

...When the change-to-LRU store instruction is executed, first, the cache is accessed using the given address. When a cache miss occurs, data is written into the main memory and the process is ended, as a store-through is assumed in the present embodiment. In the case of a cache hit, the ROW in which the hit occurs is ouputted from the selection ROW signal 117. Subsequently, when store data is written into a cache, the selection ROW signal 117 is inputted to the LRU encode circuit 110; the LRU encode circuit 110 creates a value to be written into the LRU array 111; and along with updating the data of the cache, the value of the LRU is also updated. Moreover, here, the LRU encode circuit 110 operates according to the table 303 in FIG. 3, in the same manner as during the LRU updating in the aforementioned change-to-LRU store instruction. With this, the LRU of the block in which data was written into according to the change-to-LRU store instruction becomes the block accessed in the most distant past.

FIG. 3

<Replace row determination: decoding table> 301

NEWER / ROW#	(0)	(1)	(2)	(3)	(4)	(5)
row0	0			0	0	
row1	1	0				0
row2			0	1		1
row3		1	1		1	

<Normal memory referenced LRU updating: encoding table>

NEWER / ROW#	(0)	(1)	(2)	(3)	(4)	(5)
row0	1			1	1	
row1	0	1				1
row2			1	0		0
row3		0	0		0	

302

<Change-to-LRU memory referenced LRU updating: encoding table>

NEWER / ROW#	(0)	(1)	(2)	(3)	(4)	(5)
row0	0			0	0	
row1	1	0				0
row2			0	1		1
row3		1	1		1	

303

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2001－5725
（P2001－5725A）
(43)公開日　平成13年１月12日(2001.1.12)

(51)Int.Cl.'	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０６Ｆ　12/12		Ｇ０６Ｆ　12/12	Ｄ　５Ｂ００５
			Ａ
			Ｆ

審査請求　未請求　請求項の数１　ＯＬ　（全 11 頁）

(21)出願番号　特願平11－173656

(22)出願日　平成11年６月21日(1999.6.21)

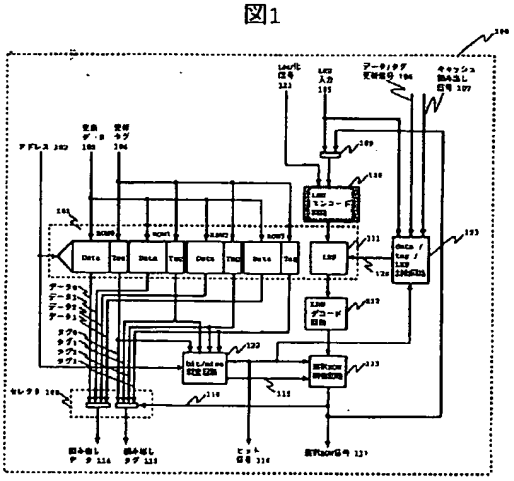
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　西本　哲
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　飯塚　孝好
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(74)代理人　100075096
弁理士　作田　康夫

最終頁に続く

(54)【発明の名称】　キャッシュ記憶装置

(57)【要約】
【課題】　セットアソシアティブ方式のキャッシュ記憶装置において、ブロック競合によるキャッシュミス回数を低減するキャッシュ記憶装置を提供すること。
【解決手段】　データとタグを保持するメモリセル部と、キャッシュブロックの参照履歴情報を表すＬＲＵメモリ部と、ヒット／ミスを判定する回路を具備するセットアソシアティブ方式のキャッシュメモリ装置において、該当キャッシュブロックのＬＲＵを最も遠い昔に参照したブロックに変更する命令を備える。前記命令の実行時に、ＬＲＵエンコード回路１１０において該当ブロックのＬＲＵを最も遠い昔に参照したブロックに変更する。

図1



1

【特許請求の範囲】

【請求項１】キャッシュブロックの追い出しアルゴリズムにＬＲＵを使用したセットアソシアティブ方式のキャッシュ記憶装置を備えるプロセッサにおいて、キャッシュブロックが最も遠い昔にアクセスされたブロックとなるように、キャッシュブロックの参照履歴情報を更新する（以後”ＬＲＵ化する”と呼ぶ）命令（以後、”ＬＲＵ化命令”と呼ぶ）を備えることを特徴とするプロセッサ。

【発明の詳細な説明】

【発明の属する技術分野】本発明はプロセッサにおけるキャッシュ記憶装置に係わり、特に、将来使用される可能性の高いデータのキャッシュからの追い出しを抑止することができるキャッシュ記憶装置に関する。

【従来の技術】キャッシュ記憶装置については、富田眞治、村上 和彰、新實 治男 訳：”コンピュータアーキテクチャ：設計・実現・評価の定量的アプローチ”：Ｐ４０３～Ｐ４２８：日経ＢＰ、において詳しく述べられている。コンピュータプログラムの主記憶参照には局所性がある。この性質を利用して、頻繁にアクセスされる主記憶データをキャッシュ記憶装置と呼ばれる高速で小容量のメモリにコピーし、主記憶へのアクセスをこのキャッシュ記憶装置に対して行なうことによって、メモリアクセスを高速化させることが行なわれている。キャッシュ記憶装置と主記憶の間のデータの管理は、ブロックと呼ぶ適当な大きさを単位として行なわれる。ブロックの大きさをブロックサイズと呼ぶ。キャッシュ記憶装置には複数のブロックが格納されており、例えばキャッシュサイズが１２８Ｋバイト、ブロックサイズが１２８バイトの場合、キャッシュ記憶装置には１０２４個のブロックが格納される。キャッシュに格納されたデータはデータアレイと呼ぶメモリに保持される。またデータと同様にキャッシュが主記憶上のどのブロックを保持するかを識別するために、ブロックのアドレスがアドレスアレイと呼ばれるメモリに保持される。プロセッサが参照するデータがキャッシュ上にあるかどうかは、アドレスアレイと、命令が参照するアドレスを比較することで判定する。主記憶上のブロックをキャッシュ上のどこに配置するかによって、キャッシュの構成は３つに分類できる。主記憶上の各ブロックのキャッシュ上での位置がそのアドレスによって一意に決められている場合をダイレクトマップ方式と呼び、主記憶上の各ブロックをキャッシュ上の任意の場所に配置して良い場合をフルアソシアティブ方式と呼び、主記憶上のブロックをキャッシュ上のある決められた範囲内の中だけに配置することができる場合を、セットアソシアティブ方式と呼ぶ。一般にフルアソシアティブ方式の場合、キャッシュ上にデータが存在するかどうかを判定するためには、命令が参照するアドレスと、キャッシュに格納されている全てのブロックのアドレスを比較しなければならず、ハードウエア量の点から現実的ではない。そのためブロックのマッピングには、従来からダイレクトマップ方式と、セットアソシアティブ方式が使用されている。セットアソシアティブ方式の概要を図４に示す。セットアソシアティブ方式では、キャッシュメモリをＮ行Ｍ列のブロックに分割する。各ブロックにはデータとブロックのアドレスが格納されている。ここでキャッシュの各行をカラムと呼び、各列をロー（ＲＯＷ）と呼ぶ。図４は、キャッシュサイズが１２８Ｋバイト、ブロックサイズが１２８バイトの４ウエイセットアソシアティブのキャッシュを表している。この例では、各カラムは４個のブロックから構成され、キャッシュは２５６個のカラムから構成される。メモリアクセスが図４の４０１、４０２、４０３に示す４０ビットのアドレスで行なわれると仮定すると、４０３の７ビットはブロック内のオフセットを表し、４０２の８ビットは、キャッシュのカラム番号を表し、４０１の２５ビットは、同一カラムに格納される可能性のある主記憶上のブロックのインデックスを表す。したがって、アドレスアレイには４０１に示すブロックのインデックスが格納される。メモリ参照命令が実行されると、まず４０２で示されるカラムアドレスによって、キャッシュがアクセスされる。ここでカラムアドレスがｉならば、キャッシュメモリのカラムｉ上の４個のブロックがアクセスされる。次に上記で読み出された４個のブロックのアドレスと、４０１で与えられるブロックのインデックスの一致を検査する。４個のいずれかのブロックとアドレスが一致していれば、キャッシュにヒットしたといい、参照するデータがキャッシュ上に存在することが分かる。４個のどのブロックとも一致しなければ、キャッシュミスしたといい、参照するデータはキャッシュ上に存在しないことが分かる。この場合、主記憶からブロックをキャッシュ上にコピーする。セットアソシアティブ方式では、主記憶からフェッチしたブロックを格納するキャッシュ上の位置の決定において、カラム番号はフェッチしたブロックのアドレスによって一意に決まるが、カラム内のどのＲＯＷにブロックを登録するかは、なんらかの方法で決定しなければならない。該当カラム内に有効でないＲＯＷが存在すれば、そのＲＯＷを置き換え対象とできるが、全てのＲＯＷが有効である場合は、カラム内のいずれかのＲＯＷを追い出して、その後、該当ＲＯＷに主記憶からのブロックをコピーする必要がある。上記の追い出しが発生する場合の、追い出し対象となるＲＯＷを決定する方法として、一般にＬＲＵ（Ｌｅａｓｔ Ｒｅｃｅｎｔｌｙ Ｕｓｅｄ）アルゴリズムが使用されている。これは、最も遠い過去に参照されたブロックを追い出し対象とする方式である。図４におけるＬＲＵ４０７は、カラム内の４個のブロックの参照履歴情報を格納するためのメモリであり、カラム毎に対応するＬＲＵ情報を保持している。このＬＲＵ情報を用いて、カラム内の４個のブロックの中で最も遠い昔に参照され

2

たブロックを検出する。

【発明が解決しようとする課題】一般にプログラムのメモリアクセスパターンを解析すると、次のような特徴が現れる場合が非常に多い。

（１）あるアドレスのデータに対するアクセスは比較的短い時間内に再発する。

（２）ある一定時間内にアクセスされるデータは比較的近いアドレスに分布する。前者は「時間的局所性」、後者は「空間的局所性」と呼ばれる。セットアソシアティブ方式において、追い出しアルゴリズムにＬＲＵを用いた場合、最も最近アクセスされたブロックは追い出されないので、メモリアクセスパターンに、”空間的局所性はあるが、時間的局所性がない”という特徴を持つ大きな配列をアクセスする場合、キャッシュ内のブロックが上記配列によって全て上書きされ、スタック上のデータなどの時間的局所性があるデータが追い出され、この追い出された時間的局所性のあるデータへの参照が再び発生した時にキャッシュミスが発生する場合がある。図５は、上記の問題が発生するプログラム例である。この例では、キャッシュサイズが１２８Ｋバイト、ブロックサイズが１２８バイトの４ウエイセットアソシアティブのキャッシュを仮定する。５０１はソーブプログラムであり、配列の各要素が８バイトである１００×１６０００の配列Ｓを１００×１６０００の配列Ｔにコピーする。内側ループ５０４で１６０００個の要素のコピーを行ない、外側ループ５０２を１００回繰り返すことで、全要素のコピーを行なう。また外側ループの各繰り返しにおいて、外側ループ５０２内には、内側ループ５０４の実行が始まる前にスタック上の局所変数へのアクセス５０３があると仮定する。また配列Ｓ、Ｔは１２８バイト境界に配置されていると仮定する。５０１のプログラムにおいて、配列Ｓ、Ｔに対するメモリ参照は、ある一定時間内に比較的近いアドレスに分布するが、同一アドレスに対する参照は一度しか発生しない。したがって、配列Ｓ、Ｔのメモリアクセスパターンには、空間的局所性はあるが、時間的局所性はないと考えられる。一方、局所変数の使用５０３で参照されるスタック上のデータは、外側ループ５０２の繰り返し毎に同一アドレスが参照されるので、時間的局所性があると考えられる。５１０は５０１の内側ループを疑似命令列で書き換えたものである。５１０の内側ループ５１３は、配列Ｓの各要素をロードして、配列Ｔの各要素にストアする疑似命令列になる。内側ループ５１３は１６０００回繰り返され、かつ配列Ｓの要素は８バイトなので、外側ループ５１１の１回の繰り返しで、キャッシュサイズと同じ１２８Ｋバイトメモリをシーケンシャルにアクセスする。このプログラムを実行すると、まず外側ループ５１１の最初の繰り返しで、局所変数の使用５１２によって、スタックのデータを格納するブロックがキャッシュ上にコピーされる。その後、内側ループ５１３の実行によって、１２８

Ｋバイトのメモリがアクセスされるため、スタックのデータを格納するキャッシュ上のブロックはキャッシュから追い出され、キャッシュ上のデータは全て配列Ｓによって上書きされてしまう。その後、外側ループ５１１の次の繰り返しの局所変数の使用５１２において、再びスタックが参照されたとき、キャッシュミスが発生する。スタック上のデータには時間的局所があるため、この例では外側ループ５１１の繰り返し毎に、局所変数の使用５１２でキャッシュミスが発生する。このような、時間的局所性のあるブロックが、空間的局所性はあるが時間的局所性のないブロックによってキャッシュから追い出されるという問題を解決するために、特開平７－２８１９５７号で示されているＬＲＵのロック方式を使用することができる。上記特許によれば、再利用される可能性のあるデータの最初の参照でLRUをロックし、該当データの最後の参照でＬＲＵのロックを解除する。この方法を、前記のプログラム例に適用すると、スタックの最初の参照でＬＲＵをロックし、関数を抜けるときにＬＲＵのロックを解除することになる。これにより、スタックのデータを保持するブロックはキャッシュ上で常に最新のままになるので、配列Ｓの参照によってスタックデータが追い出されることはなくなる。しかしながら、ＬＲＵのロック方式ではプロセスが切り替わった時にＬＲＵが動作しない可能性があり、キャッシュの使用率が低下するという問題がある。例えば、プロセスＡにおいて、前記のプログラム５１０を実行し、スタックの最初の参照でＬＲＵをロックした後で、ＬＲＵのロックが解除される前にプロセスＡがプロセスＢに切り替わった場合を考える。プロセスがＢに切り替わっても、キャッシュのＬＲＵはロックされたままなので、ロックされたカラムにおける追い出し対象のブロックは、常にＬＲＵがロックされたときに追い出し対象であったブロックとなり、プロセスＢにおける該当カラムは、ダイレクトマップのキャッシュと同様の動作になってしまい、キャッシュの使用率が著しく低下する。またこれを避けるために、プロセスがＡからＢに切り替わる時、ＯＳによってＬＲＵのロックを全て解除することも考えられるが、この処理は大きなオーバーヘッドとなる。以上のように、従来方式では、ＬＲＵによって性能が低下する場合があり、かつ前記特許に示されているＬＲＵのロック方式を使用した場合もマルチプロセス環境においてキャッシュの使用率が低下するという問題が発生する。本発明の目的は、前記のプログラム例５０１に示すような、空間的局所性はあるが、時間的局所性が無いデータの参照によって、時間的局所性のあるデータがキャッシュから追い出されることを減少させることができ、かつマルチプロセス環境においても特別な処理を行なうことなくＬＲＵを正常に動作させることができる、キャッシュ記憶装置を提供することである。

【課題を解決するための手段】本発明は、ブロックの追

い出しアルゴリズムにＬＲＵを使用するキャッシュ記憶装置において、キャッシュブロックのＬＲＵ化を指示できるメモリ参照命令を実行したとき、キャッシュ上の該当ブロックをＬＲＵ化することができる回路を備えることを特徴とする。従来、キャッシュブロックが参照された場合、該当するブロックのＬＲＵは最新となり、最も追い出されにくくなるが、本発明は、該当ブロックの最後の参照時に上記ＬＲＵ化命令を使用することで、アクセスしたキャッシュブロックを最も遠い昔に参照されたブロックにし（ブロックのＬＲＵ化）、次に同一カラムの追い出しが発生したときに、前記ＬＲＵ化したブロックを最も追い出され易くすることを特徴とする。

【発明の実施の形態】以下、図面を参照して本発明の実施例を詳細に説明する。本実施例では、以下のキャッシュ記憶装置を例に説明する。

（１）キャッシュのサイズは、１２８Ｋバイト

（２）キャッシュのブロックサイズは１２８バイト

（３）４ウエイセットアソシアティブ

（４）ブロック置き換えアルゴリズムにはＬＲＵを使用

（５）ストアスルー方式

なお、本発明は、ブロック置き換えアルゴリズムにＬＲＵを使用したセットアソシアティブ方式のキャッシュ記憶装置に適用するものあり、上記仮定におけるキャッシュサイズ、ブロックサイズ、ウエイ数が異なるキャッシュにも適用可能であり、またストアイン方式のキャッシュにも適用可能である。従来技術のセットアソシアティブ方式の説明において使用した図４は、本実施例で使用するキャッシュのメモリ部分の構造を示している。キャッシュメモリ部の動作は従来技術で説明した通りである。図１は、図４をメモリ部とする本発明のキャッシュ記憶装置の概略構成図である。１０１は図４におけるデータおよびアドレスアレイ４０６であり、１１１は図４におけるＬＲＵ４０７に対応する。本実施例のキャッシュ記憶装置は、キャッシュメモリが格納するデータおよびそのアドレスを保持するデータアレイおよびアドレスアレイ１０１と、参照履歴情報を保持するＬＲＵアレイ１１１と、読み出したデータをセレクトするセレクタ１０８と、ヒット／ミスを判定するＨＩＴ／ＭＩＳＳ判定回路１２２と、ＬＲＵアレイに書き込むＬＲＵの値を作成するＬＲＵエンコード回路１１０と、読み出したＬＲＵの値から追い出すＲＯＷを決定するＬＲＵデコード回路１１２と、データアレイおよびアドレスアレイから読み出したカラム内の４個のブロックのデータのうちの１つを選択するためのセレクト信号１１８を作成する選択ＲＯＷ制御回路１１３と、データアレイ／アドレスアレイ／ＬＲＵアレイへの書き込みおよび読み出しを制御するＤＡＴＡ／ＴＡＧ／ＬＲＵ制御回路１２３から構成される。以下では、まず通常のロード命令を実行したときの、図１のキャッシュ記憶装置の動作、特にＬＲＵデコード回路１１２とＬＲＵエンコード回路１１０の動作について説明し、その後、本発明で提案するＬＲＵ化ロードを実行したときの、図１のキャッシュ記憶装置の動作、特にＬＲＵエンコード回路１１０の動作について説明する。通常のロード命令を実行すると、命令が参照するアドレス１０２と、データおよびタグを更新するか否かを表すデータ／タグ更新信号１０６と、キャッシュからのデータの読み出しを指示するキャッシュ読み出し信号１０７が与えられる。上記信号が与えられると、まずアドレス１０２のカラムアドレス部によって、データアレイ／アドレスアレイがアクセスされ、該当カラム内のＲＯＷ０、ＲＯＷ１、ＲＯＷ２、ＲＯＷ３に格納されているデータおよびアドレスが、それぞれデータ０、１、２、３およびタグ０、１、２、３に読み出される。次にＨＩＴ／ＭＩＳＳ判定回路１２２において、読み出された４個のタグの値とロード命令が参照するアドレスのうちのブロックアドレス部分を比較する。前記の４個の比較結果のうちのいずれかが一致していれば、ロード命令が参照するブロックがキャッシュ上に存在し、ＨＩＴ／ＭＩＳＳ判定回路１２２からのヒット信号１１６が１となる。いずれのＲＯＷとも一致しなければ、ロード命令が参照するブロックはキャッシュ上には存在せず、ヒット信号１１６は０となる。ＨＩＴ／ＭＩＳＳ判定回路１２２は、上記のヒット信号１１６とともに、ヒットしたＲＯＷ番号を信号１１９に出力する。選択ＲＯＷ制御回路１１３は、ヒット／ミスに応じて、読み出すべきＲＯＷを決定する。ここでヒットならば、ヒットＲＯＷ１１９によって与えられるＲＯＷ番号を選択ＲＯＷ信号１１７に出力し、かつこれをセレクタ１０８のセレクト信号１１８に出力する。セレクタ１０８では、セレクト信号１１８を使用して、同時に読み出した４個のデータおよびタグから１つをセレクトし、キャッシュからの読み出しデータ１１４および読み出しタグ１１５をロード命令の結果として出力する。ＨＩＴ／ＭＩＳＳ判定回路１２２の結果がミスの場合、選択ＲＯＷ制御回路１１３は、ＬＲＵデコード回路１１２から出力される追い出し対象ＲＯＷを選択ＲＯＷ信号１１７に出力する。その後、主記憶からのデータのフェッチを行ない、フェッチした１ブロック分のデータをキャッシュに書き込む。ここでデータは、前述した、キャッシュミス時に選択ＲＯＷ信号１１７から出力された追い出し対象ＲＯＷに書き込む。前述のとおり、キャッシュミス時に追い出し対象となるＲＯＷは、ＬＲＵデコード回路１１２で決定される。このＬＲＵデコード回路１１２の動作を図２および図３を用いて説明する。図２は、従来から使用されている４ウエイセットアソシアティブのＬＲＵの実現方法を表している。この方法によると、４ウエイのＬＲＵを６ビットで表し、かつ最も最近アクセスされたＲＯＷの決定および、最も最近アクセスされたＲＯＷへのＬＲＵの更新を、前記６ビットのうちの３ビットだけを用いて行なうことができる。本実施例におけるＬＲＵの実現方法も、

この従来技術を使用する。図2の各ノード201、202、203、204は4ウエイセットアソシアティブにおける各ROWを表し、ノード間のエッジNEWER（0）～NEWER（5）はノード間の参照の新旧関係を表す。このエッジは有向エッジであり、矢印が指すノードは、矢印が出るノードよりも最近アクセスがあったROWであることを表す。したがって、4ノードのうちで最も遠い昔にアクセスされたノード（追い出し対象となるノード）は、どこからも矢印が指されていないノードとなる。図2に示す矢印の向きを初期状態とし、NEWER（0-5）=（0、0、0、0、0、0）とする。この場合、初期状態で最も遠い昔にアクセスされたノードはROW0である。以上のように4ウエイのLRUを実現するために、1つのカラムに対して6ビットNEWERを使用する。本実施例ではNEWERは、図1のLRUアレイ111が保持する。キャッシュミスが発生すると、該当カラム内のROWのうちで、最も遠い昔にアクセスされたROWを追い出し対象のROWとして選択する。前述のように最も遠い昔に参照されたROWは、どこからも矢印が示されていないROWである。キャッシュミス時の追い出し対象のROWは、キャッシュアクセス時にLRUアレイから読み出される該当カラムのNEWER（0-5）をLRUデコード回路112でデコードして得られるROWを使用する。図3の表301はLRUデコード回路112の制御を表している。表301はNEWER（0-5）が与えられたときに、最も遠い昔に参照されたROWを検出する論理である。例えば、ROW0が最も遠い昔にアクセスされたROWとなるのは、ROW0とつながる矢印NEWER（0）、NEWER（3）、NEWER（4）が全てROW0から出ている場合である。すなわち各矢印が初期状態と同じNEWER（0、3、4）=（0、0、0）の場合である。同様にROW1が追い出し対象となるのは、NEWER（0）だけが初期状態と異なる、NEWER（0、1、5）=（1、0、0）の場合である。以上が、通常のロード命令を実行したときの、LRUによる追い出しROWの決定方法（LRUデコード回路110の動作）である。次に通常ロード命令を実行したときの、LRUの更新動作（LRUエンコード回路112の動作）について説明する。通常のロード命令の実行では、以下の場合にLRUを更新する。

（1）キャッシュヒット時

（2）キャッシュミス時の主記憶からのデータのキャッシュへの書き込み時

（1）（2）のケースは共に該当ROWのLRUを最新（最も最近参照されたROW）に変更する。（1）のケースでは、キャッシュにヒットすると、HIT/MISS判定回路122よって出力されるヒットROW119が、選択ROW制御回路113で選択され、選択ROW信号117に出力される。その後、選択ROW信号117はLRUエンコード回路110に入力され、LRUエンコード回路110において、ROW番号がLRUアレイに登録するNEWER（0-5）にエンコードされ、LRUアレイが更新される。通常のロード命令の実行において、LRUエンコード回路110は、図3の表302にしたがって動作する。例えば、LRUエンコード回路110への入力がROW0である場合、ROW0が最新になるように、該当カラムのNEWER（0-5）を変更する。表302によると、このケースではNEWER（0、3、4）=（1、1、1、）に変更する。これは図2におけるROW0に繋がる矢印が全てROW0を向くように変更することを意味する。同様にLRUエンコード回路110への入力がROW1は、ROW1を最も最近参照したROWに変更するために、LRUアレイの該当カラムにNEWER（0、1、5）=（0、1、1）を登録する。次に（2）のキャッシュへの書き込みケースについて説明する。まずキャッシュミスが生じると、LRUアレイから該当カラムのNEWER（0-5）が出力される。これがLRUデコード回路112に入力され、最も遠い昔に参照されたROWが追い出し対象として出力される。LRUデコード回路112の処理は、前述した図3の表301にしたがって行なわれる。この追い出し対象のROWは選択ROW制御回路113に入力され、選択ROW信号として出力される。その後、主記憶からデータが戻り、キャッシュに書き込むときに、（1）のケースと同様に、前記の追い出し対象ROW番号が、LRUエンコード回路110に入力され、データを書き込むと同時に該当ROWが最も最近参照されたROWとなるように該当カラムのLRUアレイを変更する。ここでLRUエンコード回路110における処理は、（1）のケースと同様に、図3の表302に基づいて動作する。以上が通常のロード命令を実行したときのキャッシュ記憶装置の動作である。次に本発明で開示するLRU化ロード命令を実行した時のキャッシュメモリの動作について説明する。上述したように、通常のロード命令を実行すると、参照したブロックのLRUは最新になるのに対して、LRU化ロード命令を実行すると、参照したブロックのLRUは、最も遠い昔に参照したブロックになる。これにより、後続の命令で同一カラムを参照し、かつキャッシュミスが発生した場合は、LRU化ロード命令で参照され、LRU化されたROWが追い出し対象になる。LRU化ロード命令が実行されると、通常のロード命令の場合と同様に、参照するアドレスのカラム部によってデータアレイ、アドレスアレイ、LRUアレイがアクセスされ、該当するカラムのデータ、タグ、LRUが読み出される。ここでヒットすれば、ヒットしたROWのデータとタグが読み出しデータ114および読み出しタグ115によって返される。次にLRU化ロード命令の実行によるLRUアレイの更新について説明する。LRUの更新は通常ロードと同様

に、以下のケースで発生する。
（１）キャッシュヒット時
（２）キャッシュミス時の主記憶からのデータのキャッシュへの書き込み時
ＬＲＵ化ロード命令を実行するときは、（１）（２）のケースは共に該当ＲＯＷのＬＲＵを最も遠い昔に参照されたＲＯＷに変更する。（１）のヒット時には、ＨＩＴ／ＭＩＳＳ判定回路１２２からヒットしたＲＯＷ１１９が出力され、選択ＷＡＹ信号１１７に、１１９で示されるＲＯＷが出力される。その後、選択ＷＡＹ信号１１７はＬＲＵエンコード回路１１０に入力され、ＬＲＵアレイ１１１に書き込むべきＮＥＷＥＲ（０－５）の値を決定する。ＬＲＵ化ロード時は、ＬＲＵエンコード回路１１０は図３の表３０３にしたがって動作する。例えば、ＬＲＵ化ロード命令がＲＯＷ０にヒットした場合、ＬＲＵエンコード回路１１０に、ＲＯＷ０が入力される。表３０３によると、ＲＯＷ０が入力されると、ＮＥＷＥＲ（０、３、４）＝（０、０、０）が出力される。これは図２におけるノードＲＯＷ０に繋がっている矢印が、通常のロード命令とは逆に、全てＲＯＷ０から出るようにＮＥＷＥＲを変更することを意味しており、これによりＲＯＷ０は該当カラム内で最も遠い昔に参照されたＲＯＷに変更される。同様にＬＲＵ化ロード命令がＲＯＷ１にヒットした場合、ＬＲＵエンコード回路１１０にはＲＯＷ１が入力され、ＮＥＷＥＲ（０、１、５）＝（１、０、０）が出力される。これは図２におけるノードＲＯＷ１に繋がっている矢印が、通常のロード命令とは逆に、全てＲＯＷ１から出るようにＮＥＷＥＲを変更することを意味しており、これによりＲＯＷ１は該当カラム内で最も遠い昔に参照されたＲＯＷに変更される。次に（２）のキャッシュへの書き込み時の動作について説明する。ＬＲＵ化ロード命令によるキャッシュアクセスでミスが発生すると、ＨＩＴ／ＭＩＳＳ判定回路１２２からヒット信号１１６によってミスが発生したことが選択ＲＯＷ制御回路１１３に伝えられる。またＬＲＵデコード回路１１２からは、ＬＲＵ化ロード命令がアクセスしたカラム内の最も遠い昔に参照されたＲＯＷが作成され、選択ＲＯＷ制御回路１１３に入力される。選択ＲＯＷ制御回路はキャッシュミスが発生すると、選択ＲＯＷ信号１１７にＬＲＵデコード回路１１３から入力されるＲＯＷ番号を、追い出し対象ＲＯＷとして出力する。ＬＲＵ化ロード命令のキャッシュミスによって主記憶からデータがフェッチされると、そのデータをキャッシュに書き込むと同時にＬＲＵの更新も行なう。ここで新しいＬＲＵの値は、ＬＲＵ化ロード命令のミス時に追い出しＲＯＷとして選択ＲＯＷ信号１１７から出力されたＲＯＷ番号であり、この値はＬＲＵエンコード回路１１０に入力され、ＬＲＵアレイ１１１への入力信号が作成される。なお、ＬＲＵ化ロード命令による書き込み時の、ＬＲＵエンコード回路１１０は、前述したＬＲＵ化ロード命令のヒットケースと同様に、図３の表３０３にしたがって動作する。すなわち、ＬＲＵエンコード回路１１０に入力された書き込みＲＯＷのＬＲＵが、最も遠い昔にアクセスしたＲＯＷとなるように、ＬＲＵアレイへの入力信号ＮＥＷＥＲ（０－５）が作成される。上記実施例では、通常のロード命令の機能と、該当キャッシュブロックをＬＲＵ化する機能を備えるＬＲＵ化ロード命令について説明したが、請求項で示した以下の命令についても同様に実施することができる。

・ＬＲＵ化ストア命令
通常のストア命令の機能と、該当キャッシュブロックをＬＲＵ化する機能備えた命令。
・ＬＲＵ化専用命令
命令で指定したアドレスのブロックがキャッシュ上にあれば、そのブロックをＬＲＵ化するための専用命令。
・ＬＲＵ化プリフェッチ命令
通常のプリフェッチ命令の機能と、プリフェッチしたキャッシュブロックをＬＲＵ化する機能を備えた命令。
・ＬＲＵ化オフセット指定付きプリフェッチ命令
通常のプリフェッチ命令の機能と、プリフェッチアドレスから、命令で指定されたオフセットだけ離れたアドレスのブロックがキャッシュ上にあれば、このブロックをＬＲＵ化する機能を備えた命令。
・ＬＲＵ化オフセット指定付きメモリ参照命令
ロード命令、ストア命令といった通常のメモリ参照命令の機能と、メモリ参照するアドレスから、命令で指定されたオフセットだけ離れたアドレスのブロックがキャッシュ上にあれば、このブロックをＬＲＵ化する機能を備えた命令。以下では、これらの命令を実行したときの、本実施例のキャッシュ記憶装置の動作について説明する。ＬＲＵ化ストア命令を実行すると、まず与えられたアドレスで、キャッシュをアクセスし、ヒットチェックを行なう。キャッシュミスならば、本実施例はストアスルーを仮定しているので、主記憶にデータを書き込んで処理を終了する。キャッシュヒットの場合は、ヒットしたＲＯＷが選択ＲＯＷ信号１１７から出力される。その後、ストアデータをキャッシュに書き込む時に、前記選択ＲＯＷ信号１１７がＬＲＵエンコード回路１１０に入力され、ＬＲＵエンコード回路１１０によって、ＬＲＵアレイ１１１に書き込むべき値が作成され、キャッシュのデータを更新する共にＬＲＵの値も更新される。なお、ここでＬＲＵエンコード回路１１０は、前述したＬＲＵ化ロード命令におけるＬＲＵの更新時と同様に、図３の表３０３にしたがって動作する。これにより、ＬＲＵ化ストア命令によってデータが書き込まれたブロックのＬＲＵは、最も遠い昔にアクセスされたブロックとなる。ＬＲＵ化専用命令を実行すると、まず指定されたアドレスでキャッシュのヒットチェックを行なう。キャッシュミスならば処理を終了する。キャッシュヒットならば、ヒットしたＲＯＷが選択ＲＯＷ信号１１７から出力

され、ＬＲＵエンコード回路１１０に入力される。ＬＲＵエンコード回路１１０では、ＬＲＵ化ロード命令におけるＬＲＵの更新時と同様に、図３の表３０３にしたがって、更新するＬＲＵの値を生成し、ＬＲＵアレイ１１１に書き込む。これにより、ＬＲＵ化専用命令によってアクセスされたキャッシュブロックは最も遠い昔にアクセスされたブロックとなる。ＬＲＵ化プリフェッチ命令を実行すると、まずプリフェッチアドレスでキャッシュをアクセスし、ヒットチェックを行なう。キャッシュミスならば、主記憶からのデータのフェッチを行なう。ここでフェッチしたデータを書き込むキャッシュのＲＯＷは、選択ＲＯＷ信号１１７から出力される。このＲＯＷ番号は、ＬＲＵ化ロード命令におけるキャッシュミス時と同様にＬＲＵデコード回路１１２によって作成される。フェッチしたデータを書き込む時に、前記の書き込むべきＲＯＷ番号をＬＲＵエンコード回路１１０に入力し、更新するＬＲＵの値を決定する。ＬＲＵエンコード回路１１０は、ＬＲＵ化ロード命令におけるＬＲＵの更新時と同様に、図３の表３０３にしたがって動作する。これによりキャッシュに書き込んだブロックは、最も遠い昔にアクセスされたブロックとなる。なお、プリフェッチ命令では、主記憶からフェッチしたデータのレジスタへの転送は行なわない。次にＬＲＵ化プリフェッチ命令でキャッシュヒットした場合は、ＬＲＵ化専用命令でキャッシュヒットした場合と同様に動作する。アドレスアレイが２ポートのキャッシュにおいて、ＬＲＵ化オフセット指定付きプリフェッチ命令を実行すると、まず指定されたプリフェッチアドレスと、前記プリフェッチアドレスから命令で指定されたオフセット分離れたアドレス（以後、”ＬＲＵ化アドレス”と呼ぶ）によってキャッシュをアクセスし、ヒットチェックを前記の２個のアドレスで同時に行なう。プリフェッチアドレスによるキャッシュアクセスは、通常のプリフェッチ命令の動作と同様に行なう。すなわちヒットしていれば、データのレジスタへの転送は行なわず、ミスしていれば主記憶からデータをフェッチするが、レジスタへのデータの転送は行なわない。なお、プリフェッチアドレスによるキャッシュアクセスでは、アクセスしたキャッシュブロックのＬＲＵは、従来のプリフェッチ命令と同様に最新のブロックに変更する。一方ＬＲＵ化アドレスによってアクセスされたキャッシュブロックはＬＲＵ化する。このＬＲＵ化の動作は、ＬＲＵ化専用命令の動作と同様である。これにより、プリフェッチしたデータを保持するキャッシュブロックのＬＲＵは最新となり、プリフェッチアドレスとオフセットによって生成したＬＲＵ化アドレスを保持するキャッシュブロックのＬＲＵは、最も遠い昔に参照されたブロックとなる。最後に、ＬＲＵ化オフセット指定付きメモリ参照命令を、アドレスアレイが２ポートのキャッシュで実行した場合を説明する。前記命令を実行すると、まずメモリ参照するアドレスと、前記メモリ参照アドレスから命令で指定されたオフセット分離れたアドレス（ＬＲＵ化アドレス）によってキャッシュをアクセスし、ヒットチェックを前記の２個のアドレスで同時に行なう。メモリ参照アドレスによるキャッシュアクセスは、通常のメモリ参照命令と同様に行なう。すなわち、該当するキャッシュブロックのＬＲＵは最も最近参照されたブロックとなるように変更する。一方ＬＲＵ化アドレスによってアクセスされたキャッシュブロックはＬＲＵ化する。ＬＲＵ化の動作は前述したＬＲＵ化オフセット指定付きプリフェッチ命令におけるＬＲＵ化アドレスによるキャッシュアクセスと同様である。なお上記のＬＲＵ化オフセット指定付きプリフェッチ命令とＬＲＵ化オフセット指定付きメモリ参照命令の実施例では、ＬＲＵ化アドレスを、プリフェッチアドレスおよびメモリ参照アドレスからオフセット分離れたアドレスとしたが、オフセット×ブロックサイズ分離れたアドレスとする等、オフセットを用いてＬＲＵ化アドレスをエンコードすることも可能である。また、上記実施例では、２ポートのアドレスアレイを用いて、プリフェッチアドレスおよびメモリ参照アドレスと、ＬＲＵ化アドレスで同時にキャッシュをアクセスしたが、１ポートのキャッシュで実施する場合は、２度キャッシュをアクセスすることで同様の動作を実現することができる。以上、本発明で提供するキャッシュ記憶装置において、各ＬＲＵ化命令を実行したときの、キャッシュ記憶装置の動作について説明した。以下では、従来技術で問題が発生したプログラム例５０１に、ＬＲＵ化命令を適用することで、前記の問題が解決できることを示す。なお、プログラム例５０１の動作および仮定は、従来技術の説明と同じである。図５の５２０は、５１０にＬＲＵ化ロード命令およびＬＲＵ化ストア命令を適用した、変換後のプログラムである。５２０では、ＬＲＵ化命令の適用のために、まず内側ループ５１３を１６回展開して内側ループ５２３に変換する。これにより、内側ループは、１回の繰り返しでキャッシュ１ブロック分の１２８バイトのデータを参照するコードになる。次にキャッシュブロックの最後の参照命令でる５２５と５２６をＬＲＵ化命令に変換する。５２０を実行すると、内側ループのＬＲＵ化命令を実行したとき、従来は該当ブロックのＬＲＵが最新になるところが、ＬＲＵが最も遠い昔に参照された状態になり、次に同一カラムへのアクセスが発生した時には、ＬＲＵ化したブロックが追い出し対象になる。これにより、配列Ｓはキャッシュの各カラムの１つのＲＯＷを使い回すことになり、内側ループが終了したときも、局所変数が配列Ｓによって追い出されず、局所変数の使用５２２でキャッシュにヒットする。

【発明の効果】本発明によれば、命令セットにＬＲＵ化命令を設け、かつこのＬＲＵ化命令が実行されたとき、該当するキャッシュブロックのＬＲＵ情報を、最も遠い昔に参照されたブロック変更する回路をキャッシュ記憶

13

装置に具備することにより、空間的局所性はあるが、時間的局所性のないデータの参照によって、時間的局所性のあるデータがキャッシュからおいだされることを抑止することができる。またＬＲＵ化命令は、将来参照されないブロックを追い出し対象のブロックに変更する命令であり、ＬＲＵの変更をロックする方式のように、ロックを解除する必要がないため、プロセスが切り替わってもＬＲＵは動作し続け、キャッシュの使用効率を低下させることはない。

【図面の簡単な説明】

【図１】本発明の一実施例であるキャッシュ記憶装置の概略構成図である。

【図２】図１に示すキャッシュ記憶装置におけるＬＲＵ方式の追い出しアルゴリズムの実現方法を示す図である。

【図３】図１に示すＬＲＵエンコード回路およびＬＲＵデコード回路の制御方法を示す図である。

【図４】従来例および本発明のセットアソシアティブキャッシュのメモリ部の概略構成を示す図である。

【図５】従来例で性能が低下する場合のプログラム例および本発明を適用したときに生成されるプログラム例を示す図である。

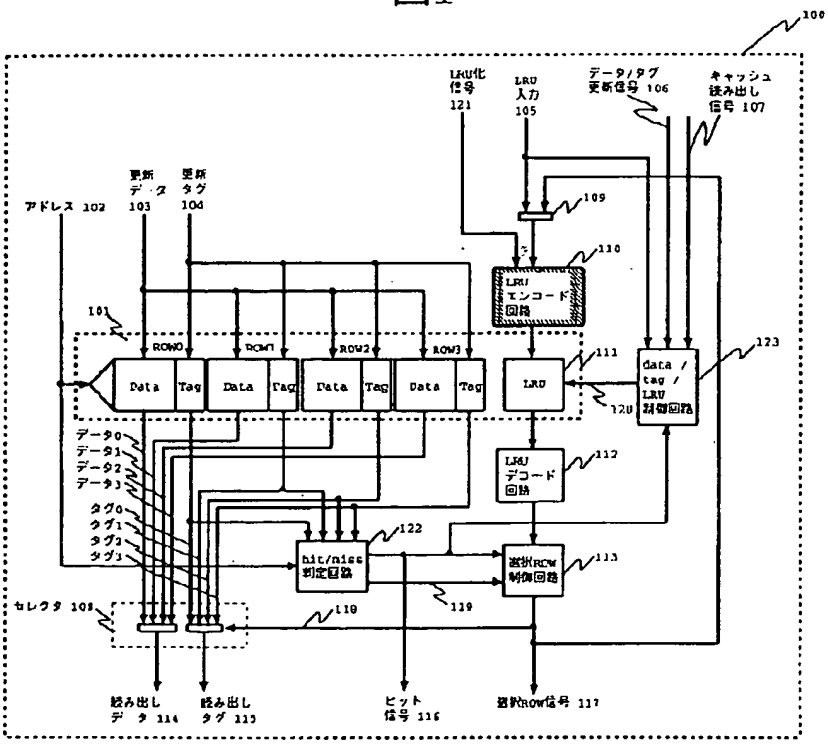
14

【符号の説明】

１００…キャッシュ記憶装置、１０１…メモリ部、１０２…キャッシュを参照するアドレス、１０３…キャッシュを更新するデータ、１０４…キャッシュを更新するタグ、１０５…更新するＬＲＵの入力、１０６…データおよびタグを更新することを伝える信号、１０７…キャッシュの読み出しを指示する信号、１０８…各ＲＯＷから読み出したデータのセレクタ、１０９…ＬＲＵ更新時のＬＲＵのセレクタ、１１０…ＬＲＵエンコード回路、１１１…ＬＲＵメモリ部、１１２…最も遠い昔に参照されたＲＯＷを検出するＬＲＵデコード回路、１１３…読み出したＬＲＵのセレクト回路、１１４…読み出しデータ、１１５…読み出しタグ、１１６…ヒット信号、１１７…選択ＲＯＷ制御回路、１１９…ＨＩＴＲＯ制御回路、１２０…データ／タグ／ＬＲＵ制御信号、１２２…ヒット／ミス判定回路、１２３…データ／タグ／ＬＲＵの制御回路。

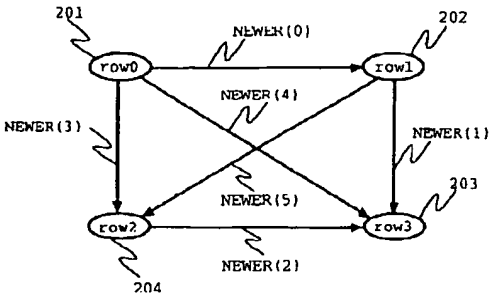
【図１】

図1



【図２】

図2



【図３】

図3

<リプレースrowの決定：デコード表>

301

NEWER / row	(0)	(1)	(2)	(3)	(4)	(5)
row0	0			0	0	
row1	1	0				0
row2			0	1		1
row3		1	1		1	

<通常のメモリ参照によるLRU更新：エンコード表>

NEWER / row	(0)	(1)	(2)	(3)	(4)	(5)
row0	1			1	1	
row1	0	1				1
row2			1	0		0
row3		0	0		0	

302

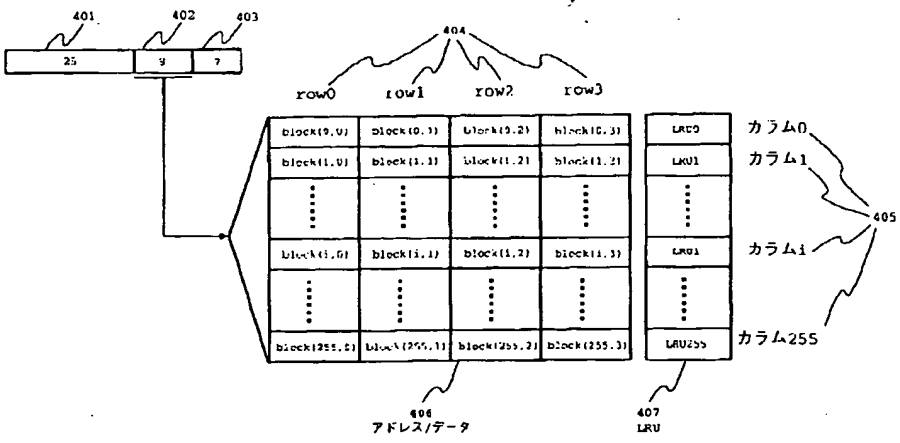
<LRU化メモリ参照によるLRU更新：エンコード表>

NEWER / row	(0)	(1)	(2)	(3)	(4)	(5)
row0	0			0	0	
row1	1	0				0
row2			0	1		1
row3		1	1		1	

303

【図４】

図4



【図５】

図5

501

[プログラム例]

```
#define  N  100
#define  M  16000
float  S[N][M];
float  T[N][M];

foo ()
{
    int  a,b,c,d;
    int  i,j;

    for ( i = 0; i < N; i ++ ) {
        :
        局所変数の使用;
        :
        for ( j = 0; j < M; j ++ ) {
            T[i][j] = S[i][j];
        }
    }
}
```

502, 503, 504

510

[疑似命令列]

```
#define  N  100
#define  M  16000
float  S[N][M];
float  T[N][M];

foo ()
{
    int  a,b,c,d;
    int  i,j;

    for ( i = 0; i < N; i ++ ) {
        :
        局所変数の使用;
        :
        for ( j = 0; j < M; j ++ ) {
            LD r1  S[i][j];
            ST r1  T[i][j];
        }
    }
}
```

511, 512, 513

520

[LRU化ロード命令の使用]

```
#define  N  100
#define  M  16000
float  S[N][M];
float  T[N][M];

foo ()
{
    int  a,b,c,d;
    int  i,j;

    for ( i = 0; i < N; i ++ ) {
        :
        局所変数の使用;
        :
        for ( j = 0; j < M; j = j + 16 ) {
            LD  r1  S[i][j];
            ST  r1  T[i][j];
            LD  r1  S[i][j+1];
            ST  r1  T[i][j+1];
            LD  r1  S[i][j+2];
            ST  r1  T[i][j+2];
            LD  r1  S[i][j+3];
            ST  r1  T[i][j+3];
            LD  r1  S[i][j+4];
            ST  r1  T[i][j+4];
            LD  r1  S[i][j+5];
            ST  r1  T[i][j+5];
            LD  r1  S[i][j+6];
            ST  r1  T[i][j+6];
            LD  r1  S[i][j+7];
            ST  r1  T[i][j+7];
            LD  r1  S[i][j+8];
            ST  r1  T[i][j+8];
            LD  r1  S[i][j+9];
            ST  r1  T[i][j+9];
            LD  r1  S[i][j+10];
            ST  r1  T[i][j+10];
            LD  r1  S[i][j+11];
            ST  r1  T[i][j+11];
            LD  r1  S[i][j+12];
            ST  r1  T[i][j+12];
            LD  r1  S[i][j+13];
            ST  r1  T[i][j+13];
            LD  r1  S[i][j+14];
            ST  r1  T[i][j+14];
            LD  LRU  r1  S[i][j+15];
            ST  LRU  r1  T[i][j+15];
        }
    }
}
```

521, 522, 523, 524, 525, 526

16回展開

LRU化ロード、ストア命令
1ライン(128バイト)中の
最後の8バイトの参照

フロントページの続き

(72)発明者　釜田　栄樹
神奈川県秦野市堀山下１番地　株式会社日立製作所エンタープライズサーバ事業部内

Ｆターム(参考)　5B005 JJ13 MM01 QQ02 TT02